

Analyses and plots for Chapter 17 – Discussion

Constituency and convergence in the Americas

Sandra Auderset

16 July, 2024

Contents

Section 3	1
Map of the sample languages	1
Dot plot of convergences and span size	4
Density plot of span size by type	6
Section 6	7
Subsection 6.2	7
Subsection 6.3	18
Section 7	23
Convergence and span size in phonological domains	23
Convergence and span size in morphosyntactic domains	25
References	26

This script provides all the code for the analyses and figures presented in the discussion chapter (17) in the order of the chapter text.

Section 3

Map of the sample languages

Create map of the sample languages. This includes all the languages which are discussed in the chapters of the volume, Chacobo, which was discussed in ([tallman2021constituency?](#)), and Siksika, which could not be included in the first volume but is fully analyzed (Natalie Weber, p.c.). The map is created using Stadia Maps ([stadiamaps.com](#)) with ggmapping Kahle & Wickham (2013).

```
# read in constituency database
domains <- read_tsv(here("domains.tsv")) %>%
  modify_if(is.character, as.factor)

## # Rows: 463 Columns: 21
## -- Column specification -----
## Delimiter: "\t"
## chr (11): Language_Name, Language_ID, Domain_ID, Domain_Type, Abstract_Type, ...
## dbl (10): Serial_Order, Left_Edge, Right_Edge, Convergence, Relative_Converg...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```

# read in metadata file
metadata <- read_tsv(here("input/metadata.tsv"))

## Rows: 37 Columns: 10
## -- Column specification -----
## Delimiter: "\t"
## chr (7): Language_Name, Alternative_Name, Short_Name, Language_ID, Glottocod...
## dbl (3): No, Latitude, Longitude
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

# subset for languages of volume 1 and Chacobo and Siksika
metadata_sub <- metadata %>%
  filter(Contribution == "Vol1" | Language_ID == "siks1238" | Language_ID == "chac1251")

# find min/max values of lat and long for map
summary(metadata_sub$Latitude)

##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
## -28.10 -12.13  14.64    9.60   18.10   60.31

summary(metadata_sub$Longitude)

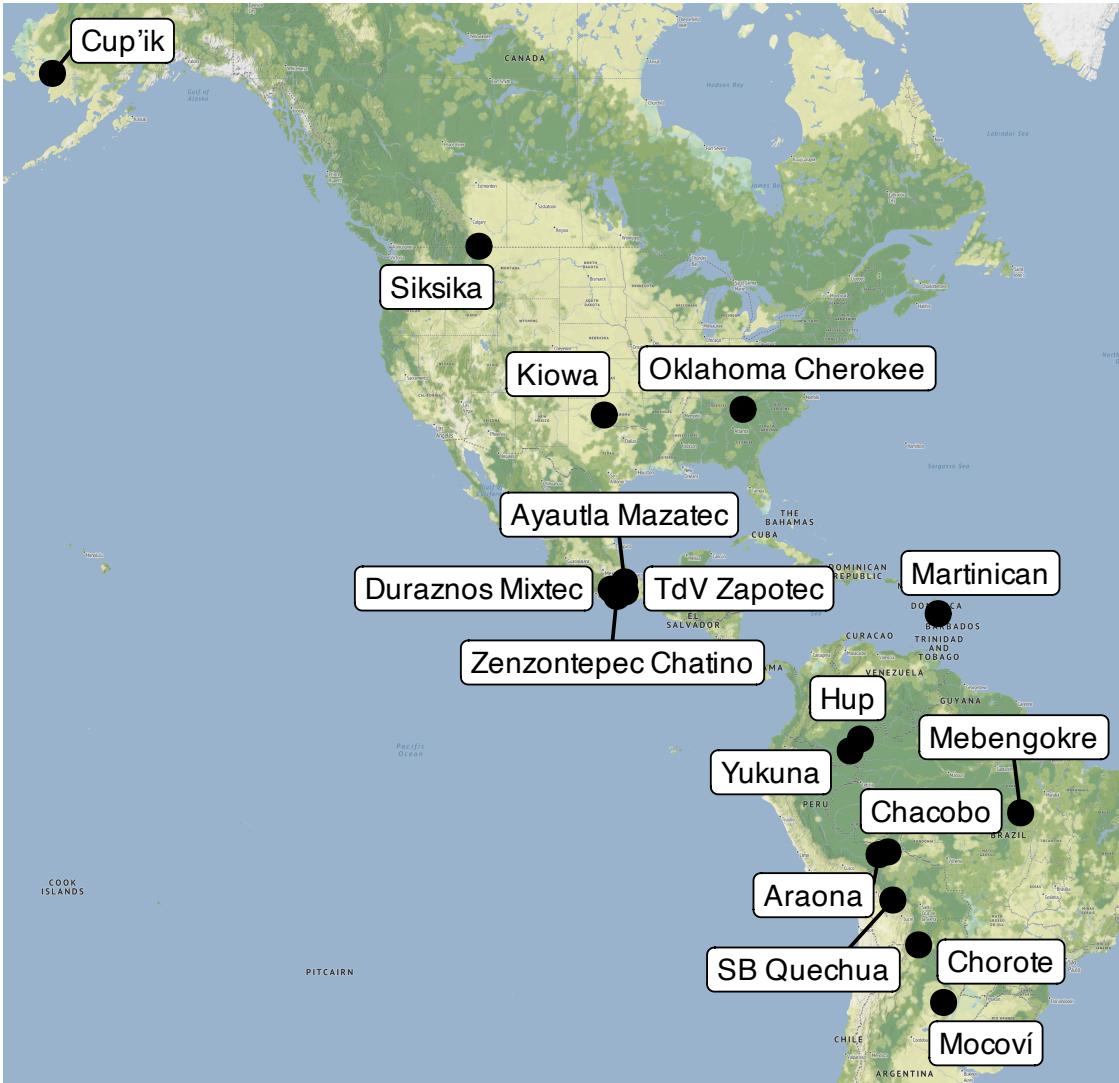
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
## -161.49 -97.46 -71.00 -83.71 -66.17 -51.67

# get basemap
americas <- get_stadiamap(bbox = c(left = -167, bottom = -36, right = -40, top = 64),
maptype = "stamen_terrain", zoom = 5)

## i © Stadia Maps © Stamen Design © OpenMapTiles © OpenStreetMap contributors.
## i 144 tiles needed, this may take a while (try a smaller zoom?)

# add points for languages
sample_map <- ggmap(americas) +
  geom_point(aes(x = Longitude, y = Latitude), data = metadata_sub, alpha = 1, size = 3)
  +
  geom_label_repel(aes(x = Longitude, y = Latitude, label = Short_Name), data =
  metadata_sub, box.padding = unit(.3, "lines"), label.padding = unit(.2, "lines"),
  max.overlaps = 30, size = 3)
  +
  theme_map()
sample_map

```



We color the points by maximum number of relative convergences per language.

```
# merge with domains file
metadata_sub_conv <- domains %>%
  select(Language_ID, Relative_Convergence) %>%
  group_by(Language_ID) %>%
  slice(which.max(Relative_Convergence)) %>%
  left_join(., metadata_sub)

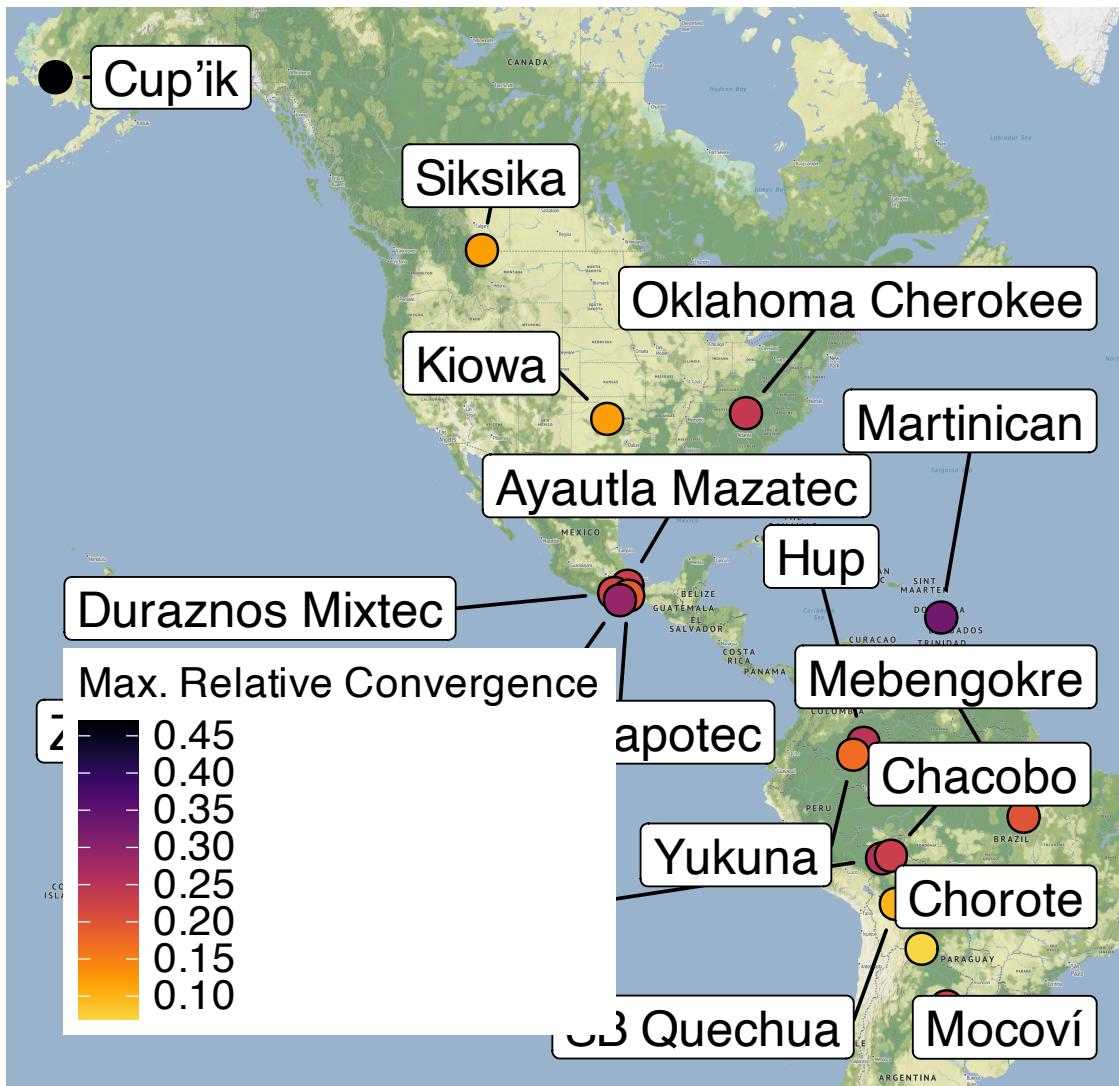
## Joining with `by = join_by(Language_ID)`

# update map
sample_map_relconv <- ggmap(americas) +
  geom_point(aes(x = Longitude, y = Latitude, fill = Relative_Convergence), data =
  metadata_sub_conv, size = 4, pch = 21) +
  scale_fill_viridis(
    option = "inferno", direction = -1, end = 0.9,
    name = "Max. Relative Convergence",
    breaks = seq(0, 0.45, by = 0.05)
  ) +
```

```

geom_label_repel(aes(x = Longitude, y = Latitude, label = Short_Name), data =
  metadata_sub, box.padding = .6, point.padding = 1, max.overlaps = 30,
  min.segment.length = 0.1, size = 5) +
  theme_map() +
  theme(
    legend.position = c(0.05, 0.05),
    legend.direction = "vertical",
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 12)
  )
sample_map_relconv

```



Dot plot of convergences and span size

We plot the relative convergences against the relative span size per sample language. We use only verbal domains as we currently have too little data for comparing nominal domains as well.

```

# exclude all domains that are not verbal; add short name of languages for plotting
domains_verbal <- domains %>%

```

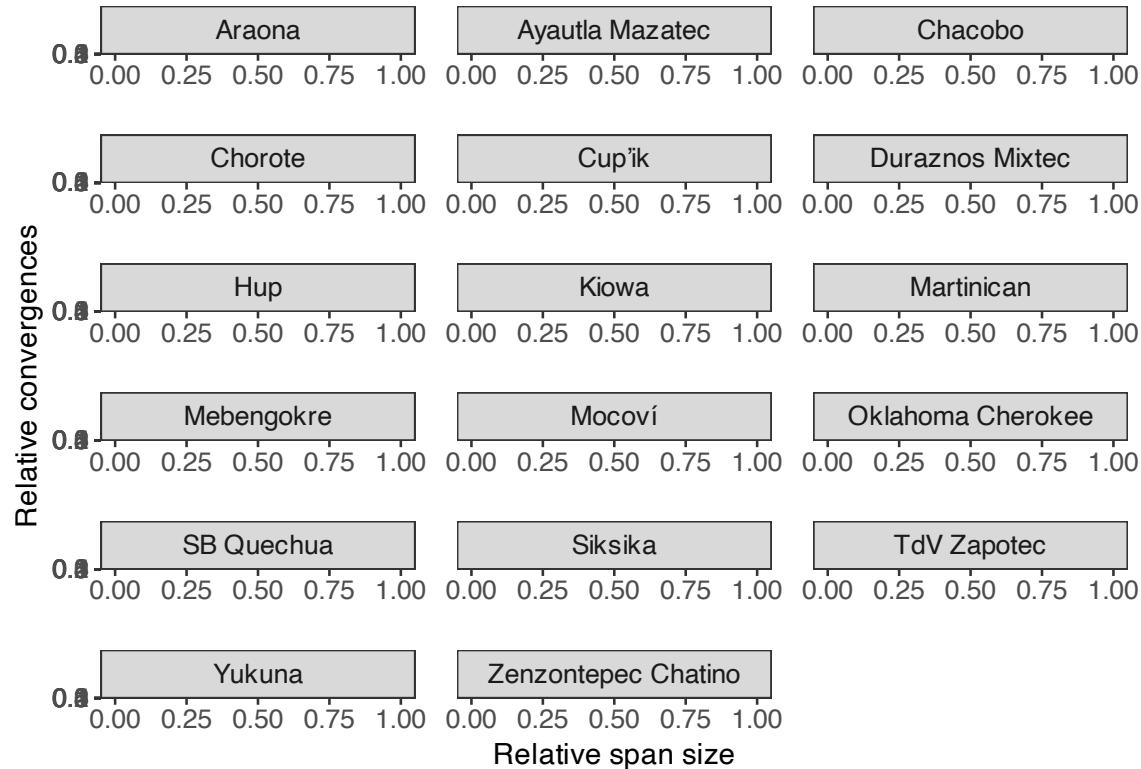
```

filter(str_detect(Planar_ID, "verbal")) %>%
filter(!is.na(Convergence)) %>%
left_join(., select(metadata_sub, Language_ID, Short_Name)) %>%
arrange(Relative_Size, Relative_Convergence)

## Joining with `by = join_by(Language_ID)`

# make dot plot per language
plot_rconv_facet <- ggplot(aes(x = Relative_Size, y = Relative_Convergence, group =
Language_Name), data = domains_verbal) +
geom_point(aes(), size = 2.5) +
geom_line(linewidth = 0.5) +
facet_wrap(~Short_Name, nrow = 6, ncol = 3, scales = "free_x") +
ylab("Relative convergences") +
xlab("Relative span size") +
scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.25)) +
scale_y_continuous(limits = c(0, 0.5)) +
theme_bw() +
theme(
  axis.text = element_text(size = 11),
  axis.title = element_text(size = 12),
  strip.text.x = element_text(size = 11),
  panel.spacing = unit(1.2, "lines")
)
plot_rconv_facet

```



Density plot of span size by type

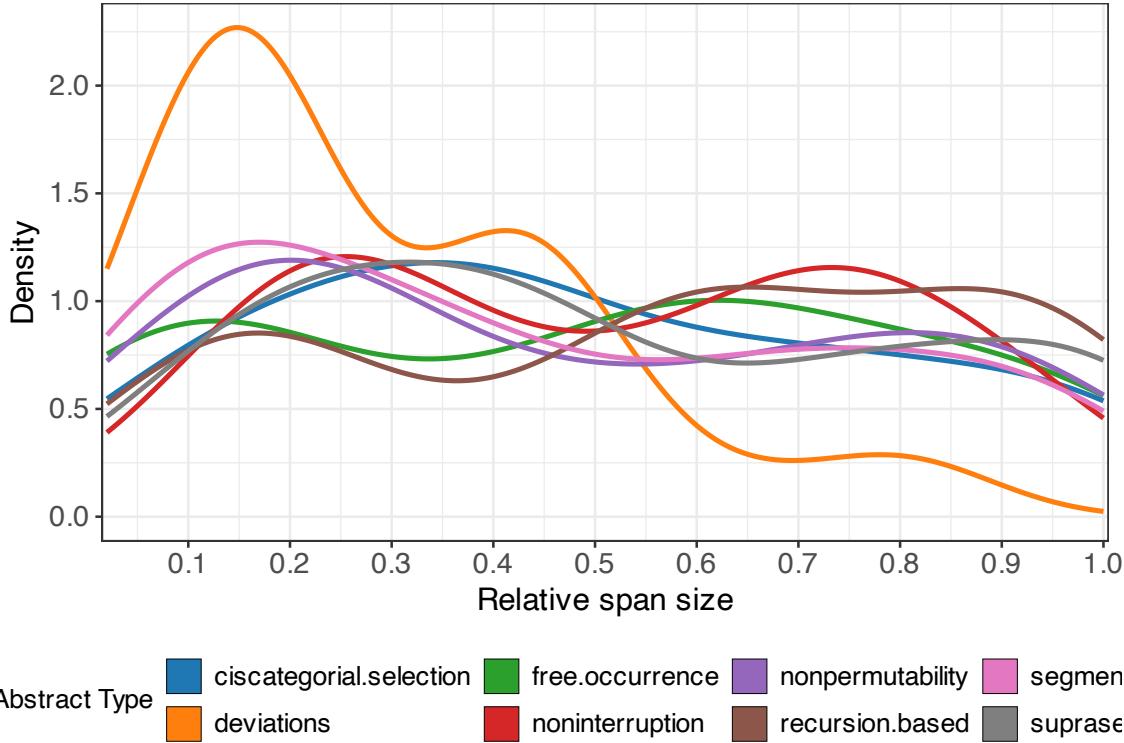
We plot densities of relative span size by abstract type of test domain. Since we are doing this across languages, we exclude abstract types that have less than 10 data points, as these are not informative.

```
# list all abstract types
levels(domains$Abstract_Type)

## [1] "ciscategorial.selection" "deviations"
## [3] "free.ocurrence"          "idiom"
## [5] "noninterruption"        "nonpermutability"
## [7] "pausing"                 "play.language"
## [9] "proform"                 "recursion.based"
## [11] "repair"                  "segmental"
## [13] "suprasegmental"

# remove types with under 10 data points
include_at <- domains_verbal %>%
  count(Abstract_Type) %>%
  arrange(desc(n)) %>%
  filter(n > 5) %>%
  droplevels() %>%
  pull(Abstract_Type)
domains_verbal_sub <- domains_verbal %>%
  filter(Abstract_Type %in% include_at) %>%
  droplevels()

# make density plot
plot_density_type <- ggplot(aes(x = Relative_Size, group = Abstract_Type), data =
domains_verbal_sub) +
  geom_density(aes(fill = Abstract_Type, color = Abstract_Type), alpha = 0, linewidth =
1) +
  scale_color_d3(palette = "category10", guide = "none") +
  scale_fill_d3(palette = "category10", name = "Abstract Type") +
  guides(fill = guide_legend(override.aes = list(alpha = 1, linewidth = 0))) +
  labs(x = "Relative span size", y = "Density", ) +
  scale_x_continuous(expand = c(0.005, 0), breaks = seq(0, 1, 0.1)) +
  theme_bw() +
  theme(
    legend.position = "bottom",
    legend.key.size = unit(1, "lines"),
    legend.text = element_text(size = 11),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 13)
  )
plot_density_type
```



Section 6

Subsection 6.2

Correlation plots across abstract types

To look at correlations between abstract types of tests, we recode all variables into binary numeric ones, where 0 indicates absence and 1 presence. We then sum these binary values per abstract type of test for each language and span. We repeat the same process, but for each language and left and right edges, respectively.

```
# recode the variables to numeric
abstract_type <- domains_verbal_sub %>%
  pivot_wider(., names_from = Abstract_Type, values_from = Abstract_Type) %>%
  mutate(across(nonpermutability:noninterruption, ~ if_else(is.na(.x), 0, 1)))
# sum overlaps by language and abstract type - for the whole span
abstract_type_span <- abstract_type %>%
  group_by(Language_ID, Left_Edge, Right_Edge) %>%
  summarise(
    Nonpermut. = sum(nonpermutability),
    Noninterrupt. = sum(noninterruption),
    Selection. = sum(ciscategorial.selection),
    Segmental. = sum(segmental),
    Free_occur. = sum(free.ocurrence),
    Recursion. = sum(recursion.based),
    Supraseg. = sum(suprasegmental),
    Deviations. = sum(deviations)
  ) %>%
  ungroup() %>%
  select(-c(Language_ID, Left_Edge, Right_Edge))
```



```

Free_occur. = sum(free.occurrence),
Recursion. = sum(recursion.based),
Supraseg. = sum(suprasegmental),
Deviations. = sum(deviations)
) %>%
ungroup() %>%
select(-c(Language_ID, Left_Edge))

## `summarise()` has grouped output by 'Language_ID'. You can override using the
## `.groups` argument.

glimpse(abstract_type_left)

## Rows: 96
## Columns: 8
## $ Nonpermut. <dbl> 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 2, 0, 0, 0, 0, 3, 0, 0, 2, ~
## $ Noninterrupt. <dbl> 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 1, ~
## $ Selection. <dbl> 1, 1, 1, 0, 0, 0, 0, 2, 0, 0, 1, 1, 0, 1, 0, 0, 2, 0, 1, ~
## $ Segmental. <dbl> 0, 2, 2, 0, 0, 0, 1, 0, 2, 1, 2, 0, 0, 0, 1, 6, 1, 0, 3, ~
## $ Free_occur. <dbl> 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 2, 0, 0, 0, 0, 1, 1, 0, 1, ~
## $ Recursion. <dbl> 1, 2, 1, 1, 2, 1, 0, 2, 0, 0, 4, 0, 0, 0, 5, 7, 0, 2, 2, ~
## $ Supraseg. <dbl> 0, 1, 1, 1, 0, 0, 0, 6, 0, 2, 2, 0, 0, 2, 1, 3, 1, 0, 0, ~
## $ Deviations. <dbl> 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, ~

```

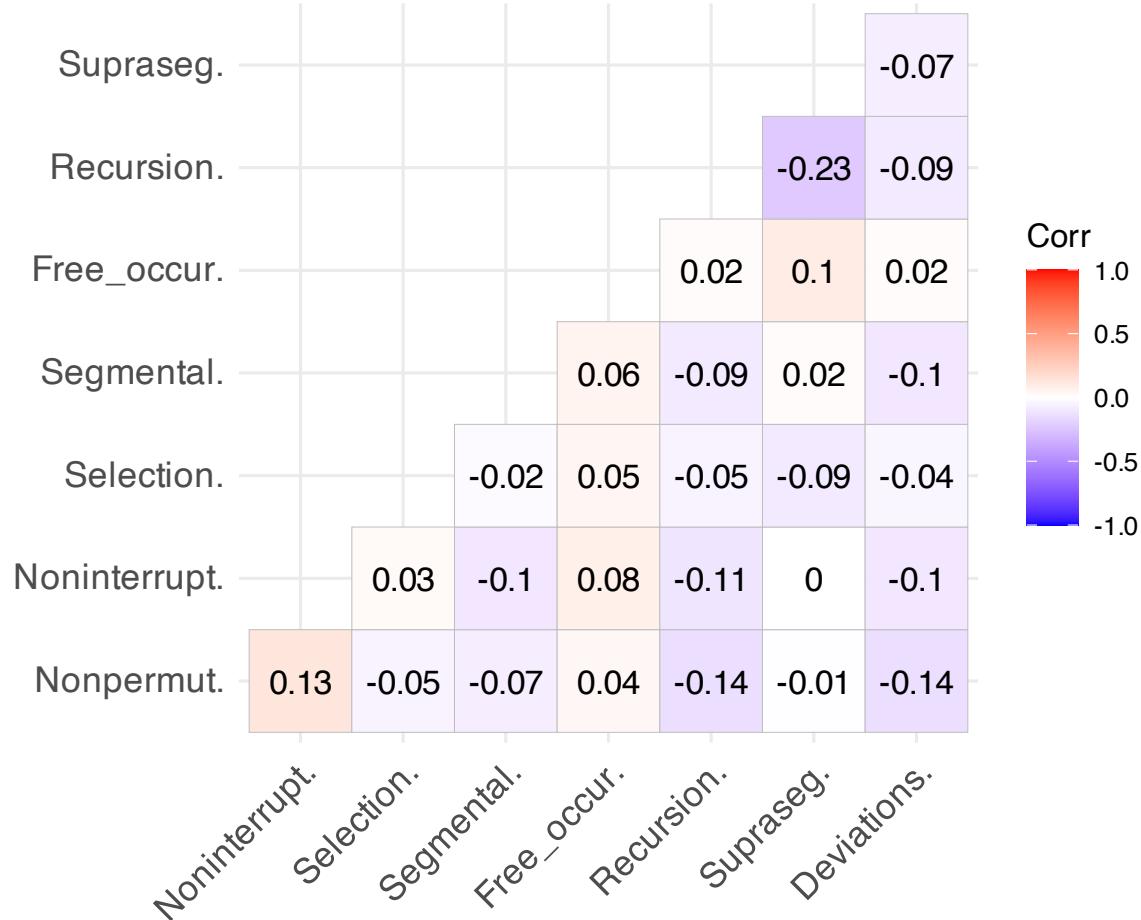
We then calculate Kendall's tau for each of these data sets and plot the results as correlation plots.

```

# make correlation matrices
corr_span <- cor(abstract_type_span, method = "kendall")
corr_left <- cor(abstract_type_left, method = "kendall")
corr_right <- cor(abstract_type_right, method = "kendall")

# plot correlations
plot_corr_span <- ggcorrplot(corr_span,
  type = "lower",
  lab = TRUE,
  lab_size = 4
)
plot_corr_span

```



```

plot_corr_left <- ggcorrplot(corr_left,
  type = "lower",
  lab = TRUE,
  lab_size = 5,
  tl.cex = 12
)

plot_corr_right <- ggcorrplot(corr_right,
  type = "lower",
  lab = TRUE,
  lab_size = 5,
  tl.cex = 12
)

```

Correlation tables for cross-language and minimal/maximal fractures

We also want to look at more fine-grained fractures. We thus create new subsets with the cross-language fractures and the minimal/maximal fractures, where available. For ciscategorial selection, free occurrence, recursion based, segmental, and suprasegmental test domains, we use the minimal/maximal fractures. For deviations from biuniqueness, non-permutability, and non-interruptability we use the cross-language fractures.

```

# data set for comparing cross-language fractures and min-max domains
cross_l <- domains_verbal_sub %>%
  mutate(MinMax_Fracture = case_match(

```

```

    MinMax_Fracture,
    "minimal" ~ "min",
    "maximal" ~ "max"
)) %>%
  mutate(abstract_cross_l = case_when(
    Abstract_Type == "deviations" ~ paste(Abstract_Type, CrossL_Fracture, sep = "_"),
    Abstract_Type == "nonpermutability" ~ paste(Abstract_Type, CrossL_Fracture, sep =
    "_"),
    Abstract_Type == "noninterruption" ~ paste(Abstract_Type, CrossL_Fracture, sep =
    "_"),
    Abstract_Type == "ciscategorial.selection" ~ paste(Abstract_Type, MinMax_Fracture,
    sep = "_"),
    Abstract_Type == "free.occurrence" ~ paste(Abstract_Type, MinMax_Fracture, sep =
    "_"),
    Abstract_Type == "recursion.based" ~ paste(Abstract_Type, MinMax_Fracture, sep =
    "_"),
    Abstract_Type == "segmental" ~ paste(Abstract_Type, MinMax_Fracture, sep = "_"),
    Abstract_Type == "suprasegmental" ~ paste(Abstract_Type, MinMax_Fracture, sep = "_")
)) %>%
  mutate(
    abstract_cross_l = str_remove_all(abstract_cross_l, "_NA"),
    abstract_cross_l = if_else(Abstract_Type == "deviations", Abstract_Type,
    abstract_cross_l)
) %>%
  arrange(abstract_cross_l) %>%
  pivot_wider(., names_from = abstract_cross_l, values_from = abstract_cross_l) %>%
  mutate(across(ciscategorial.selection:suprasegmental_min, ~ if_else(is.na(.x), 0, 1)))

```

We look at correlations between minimal and maximal domains and cross-language fractures, respectively. We do this first for the whole span and then for the right and left edge. Since there are many test pairs and we are only interested in correlations that are at least moderate, we summarize the results in tables omitting values between -0.2 and 0.2.

```

# make data set for correlations using whole span
cross_l_span <- cross_l %>%
  group_by(Language_ID, Left_Edge, Right_Edge) %>%
  summarise(
    Noninterrupt_simpl = sum(noninterruption_simplex),
    Noninterrupt_compl = sum(noninterruption_complex),
    Noninterrupt_multipos = sum(noninterruption_multipositional),
    Nonpermute_rigid = sum(nonpermutability_rigid),
    Nonpermute_scopal = sum(nonpermutability_scopal),
    Selection_min = sum(ciscategorial.selection_min),
    Selection_max = sum(ciscategorial.selection_max),
    Recursion_min = sum(recursion.based_min),
    Recursion_max = sum(recursion.based_max),
    FreeOccur_min = sum(free.occurrence_min),
    FreeOccur_max = sum(free.occurrence_max),
    Deviations = sum(deviations),
    Segmental_min = sum(segmental_min),
    Segmental_max = sum(segmental_max),
    Supraseg_min = sum(suprasegmental_min),
    Supraseg_max = sum(suprasegmental_min)
) %>%

```

```

ungroup()

## `summarise()` has grouped output by 'Language_ID', 'Left_Edge'. You can
## override using the `groups` argument.

glimpse(cross_l_span)

## Rows: 236
## Columns: 19
## $ Language_ID      <chr> "arao1248", "arao1248", "arao1248", "arao1248", ~
## $ Left_Edge        <dbl> 1, 4, 4, 4, 4, 6, 6, 6, 2, 2, 3, 3, 6, 13, ~
## $ Right_Edge       <dbl> 17, 6, 14, 15, 16, 17, 6, 11, 13, 14, 29, 31, 20 ~
## $ Noninterrupt_simpl <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Noninterrupt_compl <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ Noninterrupt_multipos <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Nonpermutable_rigid <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Nonpermutable_scopal <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ~
## $ Selection_min      <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Selection_max       <dbl> 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Recursion_min       <dbl> 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, ~
## $ Recursion_max       <dbl> 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, ~
## $ FreeOccur_min       <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ FreeOccur_max       <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Deviations           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Segmental_min        <dbl> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Segmental_max        <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Supraseg_min         <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ Supraseg_max         <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ~

# minimal domains and cross-language fractures
cross_l_min <- select(cross_l_span, -ends_with("_max"), -c(Language_ID:Right_Edge))
# maximal domains and cross-language fractures
cross_l_max <- select(cross_l_span, -ends_with("_min"), -c(Language_ID:Right_Edge))
# correlation matrices
corr_cross_l_min <- cor(cross_l_min, method = "kendall")
corr_cross_l_max <- cor(cross_l_max, method = "kendall")

# same but with edge convergences instead of whole span
# left edge:
cross_l_left <- cross_l %>%
  group_by(Language_ID, Left_Edge) %>%
  summarise(
    Noninterrupt_simpl = sum(noninterruption_simplex),
    Noninterrupt_compl = sum(noninterruption_complex),
    Noninterrupt_multipos = sum(noninterruption_multipositional),
    Nonpermutable_rigid = sum(nonpermutable_rigid),
    Nonpermutable_scopal = sum(nonpermutable_scopal),
    Selection_min = sum(ciscategorial.selection_min),
    Selection_max = sum(ciscategorial.selection_max),
    Recursion_min = sum(recursion.based_min),
    Recursion_max = sum(recursion.based_max),
    FreeOccur_min = sum(free.occurrence_min),
    FreeOccur_max = sum(free.occurrence_max),
    Deviations = sum(deviations),

```

```

Segmental_min = sum(segmental_min),
Segmental_max = sum(segmental_max),
Supraseg_min = sum(suprasegmental_min),
Supraseg_max = sum(suprasegmental_min)
) %>%
ungroup()

# right edge:
cross_l_right <- cross_l %>%
  group_by(Language_ID, Right_Edge) %>%
  summarise(
    Noninterrupt_simpl = sum(noninterruption_simplex),
    Noninterrupt_compl = sum(noninterruption_complex),
    Noninterrupt_multipos = sum(noninterruption_multipositional),
    Nonpermut_rigid = sum(nonpermutability_rigid),
    Nonpermut_scopal = sum(nonpermutability_scopal),
    Selection_min = sum(ciscategorial.selection_min),
    Selection_max = sum(ciscategorial.selection_max),
    Recursion_min = sum(recursion.based_min),
    Recursion_max = sum(recursion.based_max),
    FreeOccur_min = sum(free.occurrence_min),
    FreeOccur_max = sum(free.occurrence_max),
    Deviations = sum(deviations),
    Segmental_min = sum(segmental_min),
    Segmental_max = sum(segmental_max),
    Supraseg_min = sum(suprasegmental_min),
    Supraseg_max = sum(suprasegmental_min)
) %>%
ungroup()

# minimal domains and cross-language fractures - left edges
cross_l_min_left <- select(cross_l_left, -ends_with("_max"), -c(Language_ID:Left_Edge))
# maximal domains and cross-language fractures
cross_l_max_left <- select(cross_l_left, -ends_with("_min"), -c(Language_ID:Left_Edge))

# minimal domains and cross-language fractures - right edges
cross_l_min_right <- select(cross_l_right, -ends_with("_max"),
-c(Language_ID:Right_Edge))
# maximal domains and cross-language fractures
cross_l_max_right <- select(cross_l_right, -ends_with("_min"),
-c(Language_ID:Right_Edge))

```

To create the correlation tables efficiently, we set up a custom function.

```

# set up function to create correlation dataframes
corr_to_df <- function(x, y, z, a, b, c) {
  df1 <- cor(x, method = "kendall") %>%
    as.data.frame() %>%
    rownames_to_column(., var = "Test1") %>%
    pivot_longer(-Test1, names_to = "Test2", values_to = paste0(y, ".", z))
  df2 <- cor(a, method = "kendall") %>%
    as.data.frame() %>%
    rownames_to_column(., var = "Test1") %>%
    pivot_longer(-Test1, names_to = "Test2", values_to = paste0(b, ".", c)) %>%

```

```

    select(-c(Test1, Test2))
# full correlation df
corr_full <- bind_cols(df1, df2) %>%
  distinct(across(starts_with("M")), .keep_all = TRUE) %>%
  filter(Test1 != Test2) %>%
  kable(
    booktabs = TRUE, linesep = "",
    align = "llrr"
  )
# excluding weak correlations and highlighting cells with higher correlations;
# exporting table to latex
corr_higher <- bind_cols(df1, df2) %>%
  distinct(across(starts_with("M")), .keep_all = TRUE) %>%
  filter(Test1 != Test2) %>%
  mutate(across(starts_with("M"), ~ round(.x, 2))) %>%
  mutate(across(starts_with("M"), ~ case_when(
    .x > 0.3 ~ paste0("\\cellcolor{red!45}", .x),
    between(.x, 0.2, 0.3) ~ paste0("\\cellcolor{red!25}", .x),
    .x < -0.3 ~ paste0("\\cellcolor{blue!45}", .x),
    between(.x, -0.2, 0.3) ~ paste0("\\cellcolor{blue!25}", .x),
    .default = paste(.x)
  ))) %>%
  mutate(across(starts_with("Test"), ~ str_replace_all(.x, "_", ".") )) %>%
  filter(if_any(starts_with("M"), ~ str_detect(., "cellcolor")))) %>%
  kable(
    booktabs = TRUE, format = "latex", escape = FALSE, linesep = "",
    align = "llrr"
  ) %>%
  kable_styling()
# return full df and latex table
return(list(corr_full, corr_higher))
}

```

We apply the function to each subset and display the full correlation tables. The highlighted tables are exported for inclusion in the chapter.

```

# cross-language + min/max - whole span
corr_cross_l_span <- corr_to_df(cross_l_min, "Min", "Span", cross_l_max, "Max", "Span")
corr_cross_l_span[[1]]

```

Test1	Test2	Min.Span	Max.Span
Noninterrupt_simpl	Noninterrupt_compl	-0.0350877	-0.0350877
Noninterrupt_simpl	Noninterrupt_multipos	-0.0245959	-0.0245959
Noninterrupt_simpl	Nonpermut_rigid	0.1431491	0.1431491
Noninterrupt_simpl	Nonpermut_scopal	-0.0327498	-0.0327498
Noninterrupt_simpl	Selection_min	-0.0470398	0.0383744
Noninterrupt_simpl	Recursion_min	0.0910499	-0.0681671
Noninterrupt_simpl	FreeOccur_min	0.0306309	0.2027476
Noninterrupt_simpl	Deviations	0.0024339	0.0024339
Noninterrupt_simpl	Segmental_min	0.0860252	-0.0596055
Noninterrupt_simpl	Supraseg_min	0.0085362	0.0085362
Noninterrupt_compl	Nonpermut_rigid	-0.0488008	-0.0488008
Noninterrupt_compl	Selection_min	-0.0470398	-0.0521891

Test1	Test2	Min.Span	Max.Span
Noninterrupt_compl	Recursion_min	0.0140833	-0.0681671
Noninterrupt_compl	FreeOccur_min	-0.0554274	-0.0554274
Noninterrupt_compl	Deviations	-0.0681498	-0.0681498
Noninterrupt_compl	Segmental_min	0.0109136	-0.0596055
Noninterrupt_compl	Supraseg_min	0.0828325	0.0828325
Noninterrupt_multipos	Nonpermut_rigid	-0.0342086	-0.0342086
Noninterrupt_multipos	Nonpermut_scopal	-0.0229571	-0.0229571
Noninterrupt_multipos	Selection_min	0.2449508	-0.0365838
Noninterrupt_multipos	Recursion_min	-0.0440803	-0.0477841
Noninterrupt_multipos	FreeOccur_min	-0.0388537	0.0817974
Noninterrupt_multipos	Deviations	-0.0477720	-0.0477720
Noninterrupt_multipos	Segmental_min	-0.0450017	-0.0417825
Noninterrupt_multipos	Supraseg_min	-0.0460968	-0.0460968
Nonpermut_rigid	Nonpermut_scopal	-0.0455492	-0.0455492
Nonpermut_rigid	Selection_min	0.0081001	-0.0054084
Nonpermut_rigid	Recursion_min	0.0267238	-0.0948084
Nonpermut_rigid	FreeOccur_min	0.0505817	0.0505817
Nonpermut_rigid	Deviations	-0.0947844	-0.0947844
Nonpermut_rigid	Segmental_min	0.0280959	-0.0829007
Nonpermut_rigid	Supraseg_min	-0.0363498	-0.0363498
Nonpermut_scopal	Selection_min	-0.0439055	-0.0487117
Nonpermut_scopal	Recursion_min	-0.0586934	-0.0636251
Nonpermut_scopal	FreeOccur_min	-0.0517343	0.1318640
Nonpermut_scopal	Deviations	-0.0636090	-0.0636090
Nonpermut_scopal	Segmental_min	-0.0599204	-0.0556340
Nonpermut_scopal	Supraseg_min	-0.0613785	-0.0613785
Selection_min	Recursion_min	0.0336211	0.0468614
Selection_min	FreeOccur_min	0.1234739	0.0380306
Selection_min	Deviations	0.0230740	-0.0519604
Selection_min	Segmental_min	0.0290165	-0.0886566
Selection_min	Supraseg_min	0.0825896	0.0061962
Recursion_min	FreeOccur_min	0.2590134	-0.0137639
Recursion_min	Deviations	0.0087240	-0.1323989
Recursion_min	Segmental_min	0.1131364	-0.0259420
Recursion_min	Supraseg_min	-0.0736586	-0.0872182
FreeOccur_min	Deviations	0.0910615	-0.0607077
FreeOccur_min	Segmental_min	0.2982587	-0.0418979
FreeOccur_min	Supraseg_min	0.0937864	-0.0050468
Deviations	Segmental_min	0.0488071	-0.1157700
Deviations	Supraseg_min	0.0391242	0.0391242
Segmental_min	Supraseg_min	0.1863994	-0.0164050

```
# cross-language + minimal - left/right
corr_cross_l_min <- corr_to_df(cross_l_min_left, "Min", "Left", cross_l_min_right, "Min",
"Right")
corr_cross_l_min[[1]]
```

Test1	Test2	Min.Left	Min.Right
Noninterrupt_simpl	Noninterrupt_compl	0.0586075	0.2041985
Noninterrupt_simpl	Noninterrupt_multipos	0.1538093	-0.0425375

Test1	Test2	Min.Left	Min.Right
Noninterrupt_simpl	Nonpermut_rigid	0.3817291	0.1131670
Noninterrupt_simpl	Nonpermut_scopal	0.0734833	0.0843454
Noninterrupt_simpl	Selection_min	0.2571236	-0.0827024
Noninterrupt_simpl	Recursion_min	0.0884481	0.2159912
Noninterrupt_simpl	FreeOccur_min	-0.0396129	-0.0039560
Noninterrupt_simpl	Deviations	0.2272164	0.0413788
Noninterrupt_simpl	Segmental_min	0.2919756	0.0507138
Noninterrupt_simpl	Supraseg_min	0.1680648	-0.0320786
Noninterrupt_compl	Noninterrupt_multipos	-0.0628695	0.1422349
Noninterrupt_compl	Nonpermut_rigid	0.1194933	-0.0859497
Noninterrupt_compl	Nonpermut_scopal	0.2053565	-0.0569077
Noninterrupt_compl	Selection_min	0.0981295	-0.0827024
Noninterrupt_compl	Recursion_min	0.4573045	0.0534044
Noninterrupt_compl	FreeOccur_min	0.1340079	-0.0039560
Noninterrupt_compl	Deviations	0.0763835	0.0540223
Noninterrupt_compl	Segmental_min	0.2138178	0.0642767
Noninterrupt_compl	Supraseg_min	0.2277638	0.1277312
Noninterrupt_multipos	Nonpermut_rigid	0.0729153	0.0788268
Noninterrupt_multipos	Nonpermut_scopal	0.1420172	0.1571412
Noninterrupt_multipos	Selection_min	0.3882387	0.2283690
Noninterrupt_multipos	Recursion_min	-0.0947373	0.0371990
Noninterrupt_multipos	FreeOccur_min	-0.1035780	0.0606219
Noninterrupt_multipos	Deviations	0.0236155	0.0464363
Noninterrupt_multipos	Segmental_min	0.0442955	0.0542194
Noninterrupt_multipos	Supraseg_min	0.0158837	0.0333135
Nonpermut_rigid	Nonpermut_scopal	0.2577841	0.2379919
Nonpermut_rigid	Selection_min	0.1995048	0.1147347
Nonpermut_rigid	Recursion_min	0.1786094	0.0904180
Nonpermut_rigid	FreeOccur_min	0.0414455	0.1484730
Nonpermut_rigid	Deviations	0.0140816	0.1419272
Nonpermut_rigid	Segmental_min	0.1165267	-0.0230173
Nonpermut_rigid	Supraseg_min	0.3614843	0.0223279
Nonpermut_scopal	Selection_min	0.1200822	0.0322425
Nonpermut_scopal	Recursion_min	0.0910146	-0.0151667
Nonpermut_scopal	FreeOccur_min	-0.0387557	0.0084261
Nonpermut_scopal	Deviations	-0.0640026	-0.1028252
Nonpermut_scopal	Segmental_min	0.0650839	-0.0163285
Nonpermut_scopal	Supraseg_min	0.1719241	-0.1043535
Selection_min	Recursion_min	0.1302715	0.0408683
Selection_min	FreeOccur_min	0.2556631	0.3643023
Selection_min	Deviations	0.1775783	0.0982878
Selection_min	Segmental_min	0.1876479	0.0378763
Selection_min	Supraseg_min	0.2406169	0.1060675
Recursion_min	FreeOccur_min	0.3116368	0.2927496
Recursion_min	Deviations	0.1809762	0.0636153
Recursion_min	Segmental_min	0.3039147	0.1155138
Recursion_min	Supraseg_min	0.2436265	-0.0043524
FreeOccur_min	Deviations	0.2231660	0.2946544
FreeOccur_min	Segmental_min	0.4292868	0.4054280
FreeOccur_min	Supraseg_min	0.3697661	0.2289099
Deviations	Segmental_min	0.3843538	0.1275054
Deviations	Supraseg_min	0.2306674	0.1510526

Test1	Test2	Min.Left	Min.Right
Segmental_min	Supraseg_min	0.3617206	0.2278026

```
# cross-language + maximal - left/right
corr_cross_l_max <- corr_to_df(cross_l_max_left, "Max", "Left", cross_l_max_right, "Max",
"Right")
corr_cross_l_max[[1]]
```

Test1	Test2	Max.Left	Max.Right
Noninterrupt_simpl	Noninterrupt_compl	0.0586075	0.2041985
Noninterrupt_simpl	Noninterrupt_multipos	0.1538093	-0.0425375
Noninterrupt_simpl	Nonpermut_rigid	0.3817291	0.1131670
Noninterrupt_simpl	Nonpermut_scopal	0.0734833	0.0843454
Noninterrupt_simpl	Selection_max	0.1882274	0.0020349
Noninterrupt_simpl	Recursion_max	-0.0520398	0.0467851
Noninterrupt_simpl	FreeOccur_max	0.2574835	0.2774557
Noninterrupt_simpl	Deviations	0.2272164	0.0413788
Noninterrupt_simpl	Segmental_max	0.2722157	-0.0079399
Noninterrupt_simpl	Supraseg_max	0.1680648	-0.0320786
Noninterrupt_compl	Noninterrupt_multipos	-0.0628695	0.1422349
Noninterrupt_compl	Nonpermut_rigid	0.1194933	-0.0859497
Noninterrupt_compl	Nonpermut_scopal	0.2053565	-0.0569077
Noninterrupt_compl	Selection_max	0.0575922	0.0020349
Noninterrupt_compl	Recursion_max	0.1677723	-0.1076057
Noninterrupt_compl	FreeOccur_max	0.2286017	0.0929342
Noninterrupt_compl	Deviations	0.0763835	0.0540223
Noninterrupt_compl	Segmental_max	0.1244123	-0.0079399
Noninterrupt_compl	Supraseg_max	0.2277638	0.1277312
Noninterrupt_multipos	Nonpermut_rigid	0.0729153	0.0788268
Noninterrupt_multipos	Nonpermut_scopal	0.1420172	0.1571412
Noninterrupt_multipos	Selection_max	0.1763842	-0.0642551
Noninterrupt_multipos	Recursion_max	-0.1166327	-0.0749530
Noninterrupt_multipos	FreeOccur_max	0.0272574	0.0647335
Noninterrupt_multipos	Deviations	0.0236155	0.0464363
Noninterrupt_multipos	Segmental_max	0.1642564	-0.0714714
Noninterrupt_multipos	Supraseg_max	0.0158837	0.0333135
Nonpermut_rigid	Nonpermut_scopal	0.2577841	0.2379919
Nonpermut_rigid	Selection_max	0.0737923	0.0117103
Nonpermut_rigid	Recursion_max	-0.0565064	-0.0355571
Nonpermut_rigid	FreeOccur_max	0.5241642	0.0096045
Nonpermut_rigid	Deviations	0.0140816	0.1419272
Nonpermut_rigid	Segmental_max	0.1826726	-0.0733522
Nonpermut_rigid	Supraseg_max	0.3614843	0.0223279
Nonpermut_scopal	Selection_max	-0.0251447	-0.0859620
Nonpermut_scopal	Recursion_max	0.0326811	-0.1002739
Nonpermut_scopal	FreeOccur_max	0.2629103	0.1111662
Nonpermut_scopal	Deviations	-0.0640026	-0.1028252
Nonpermut_scopal	Segmental_max	0.1522914	-0.0956161
Nonpermut_scopal	Supraseg_max	0.1719241	-0.1043535
Selection_max	Recursion_max	0.0839477	0.0773226
Selection_max	FreeOccur_max	0.2489805	0.1321240

Test1	Test2	Max.Left	Max.Right
Selection_max	Deviations	0.0845809	0.1147964
Selection_max	Segmental_max	-0.0293431	0.0468888
Selection_max	Supraseg_max	0.0464228	0.1716444
Recursion_max	FreeOccur_max	0.0243731	0.0160968
Recursion_max	Deviations	0.0755752	-0.0961583
Recursion_max	Segmental_max	0.0442958	0.0524054
Recursion_max	Supraseg_max	0.0970535	-0.0428956
FreeOccur_max	Deviations	-0.0311684	-0.0073009
FreeOccur_max	Segmental_max	0.0614584	0.1146591
FreeOccur_max	Supraseg_max	0.2781544	-0.0498077
Deviations	Segmental_max	0.1460861	-0.0838708
Deviations	Supraseg_max	0.2306674	0.1510526
Segmental_max	Supraseg_max	0.2952724	-0.0235193

Subsection 6.3

Density plot of relative convergences across span and edges

In order to assess if there are overall tendencies for some domains to converge than others, we explore the density distributions of span and edge convergences pooled across languages of the sample. We do this by calculating the convergences across the span, and on the right and left edge and displaying the densities of these distributions.

```
# calculate right and left edge convergences
domains_verbal <- domains_verbal %>%
  group_by(Language_ID, Left_Edge) %>%
  mutate(Convergence_Left = n()) %>%
  group_by(Language_ID, Right_Edge) %>%
  mutate(Convergence_Right = n()) %>%
  ungroup() %>%
  mutate(Relative_Convergence_Left = round(Convergence_Left / Tests_Total, 2),
  Relative_Convergence_Right = round(Convergence_Right / Tests_Total, 2))
glimpse(domains_verbal)
```

```
## Rows: 388
## Columns: 26
## $ Serial_Order          <dbl> 366, 239, 224, 25, 40, 44, 178, 180, 182, 1~
## $ Language_Name          <fct> South Bolivian Quechua, Chorote, Hup, Ayaut~
## $ Language_ID             <chr> "sout2991", "iyoj1235", "hupd1244", "ayau12~
## $ Domain_ID               <fct> "sout2991_v_nonpermutability_minimal_chi.(1~
## $ Domain_Type              <fct> morphosyntactic, indeterminate, morphosynta~
## $ Abstract_Type            <fct> nonpermutability, free_occurrence, recursio~
## $ CrossL_Fracture          <fct> NA, NA, NA, NA, vowel-consonant.dissimilati~
## $ MinMax_Fracture          <fct> minimal, minimal, NA, minimal, minimal, min~
## $ Lspecific_Fracture        <fct> "chi.(13).and.chi.(18).not.one.morpheme", N~
## $ Left_Edge                 <dbl> 12, 16, 6, 19, 19, 19, 17, 17, 17, 17, ~
## $ Right_Edge                <dbl> 12, 16, 6, 19, 19, 19, 17, 17, 17, 17, ~
## $ Convergence                <dbl> 1, 1, 1, 3, 3, 3, 6, 6, 6, 6, 1, 1, 1~
## $ Relative_Convergence       <dbl> 0.05, 0.04, 0.07, 0.10, 0.10, 0.10, 0.22, 0~
## $ Size                      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2~
## $ Relative_Size              <dbl> 0.02, 0.03, 0.03, 0.03, 0.03, 0.03, 0.03, 0~
## $ Largest                    <dbl> 42, 37, 32, 30, 30, 29, 29, 29, 29, 29, 29, ~
## $ Position_Total              <dbl> 42, 46, 32, 31, 31, 29, 29, 29, 29, 29, 29, ~
## $ Tests_Total                  <dbl> 20, 28, 15, 30, 30, 27, 27, 27, 27, 27, ~
```

```

## $ Planar_ID <fct> sout2991_verbal, iyoj1235_verbal, hupd1244_~  

## $ PW_Pattern <fct> morphosyntactic, indeterminate, morphosynta~  

## $ Test_Labels <fct> Non-permut., NA, Subspan Rep. Base, Min Fre~  

## $ Short_Name <chr> "SB Quechua", "Chorote", "Hup", "Ayautla Ma~  

## $ Convergence_Left <int> 16, 6, 4, 4, 4, 4, 8, 8, 8, 8, 8, 3, 1, ~  

## $ Convergence_Right <int> 1, 4, 1, 9, 9, 9, 8, 8, 8, 8, 8, 2, 4, 1~  

## $ Relative_Convergence_Left <dbl> 0.80, 0.21, 0.27, 0.13, 0.13, 0.13, 0.30, 0~  

## $ Relative_Convergence_Right <dbl> 0.05, 0.14, 0.07, 0.30, 0.30, 0.30, 0.30, 0~  

# data set for plotting density  

domains_verbal_density <- domains_verbal %>%  

  rename(  

    Span = Relative_Convergence,  

    Left = Relative_Convergence_Left,  

    Right = Relative_Convergence_Right  

) %>%  

  distinct(Language_ID, Span, Right, Left) %>%  

  pivot_longer(-Language_ID, names_to = "Convergence_Type", values_to =  

  "Relative_Convergence") %>%  

  mutate(Convergence_Type = factor(Convergence_Type, levels = c("Span", "Left",  

  "Right")))  

# density plot across all languages  

plot_density_all <- ggplot(data = domains_verbal_density, aes(x = Relative_Convergence,  

group = Convergence_Type, fill = Convergence_Type)) +  

  geom_density(alpha = 0.6) +  

  scale_fill_d3(name = "Type of Convergence") +  

  # guides(fill = guide_legend	override.aes = list(alpha = 1, linewidth = 0)) +  

  labs(x = "Relative convergence", y = "Density", ) +  

  scale_x_continuous(expand = c(0.005, 0.005), breaks = seq(0, 1, 0.2)) +  

  theme_bw() +  

  theme(  

    legend.position = c(0.85, 0.85),  

    # legend.key.size = unit(1, "lines"),  

    legend.text = element_text(size = 11),  

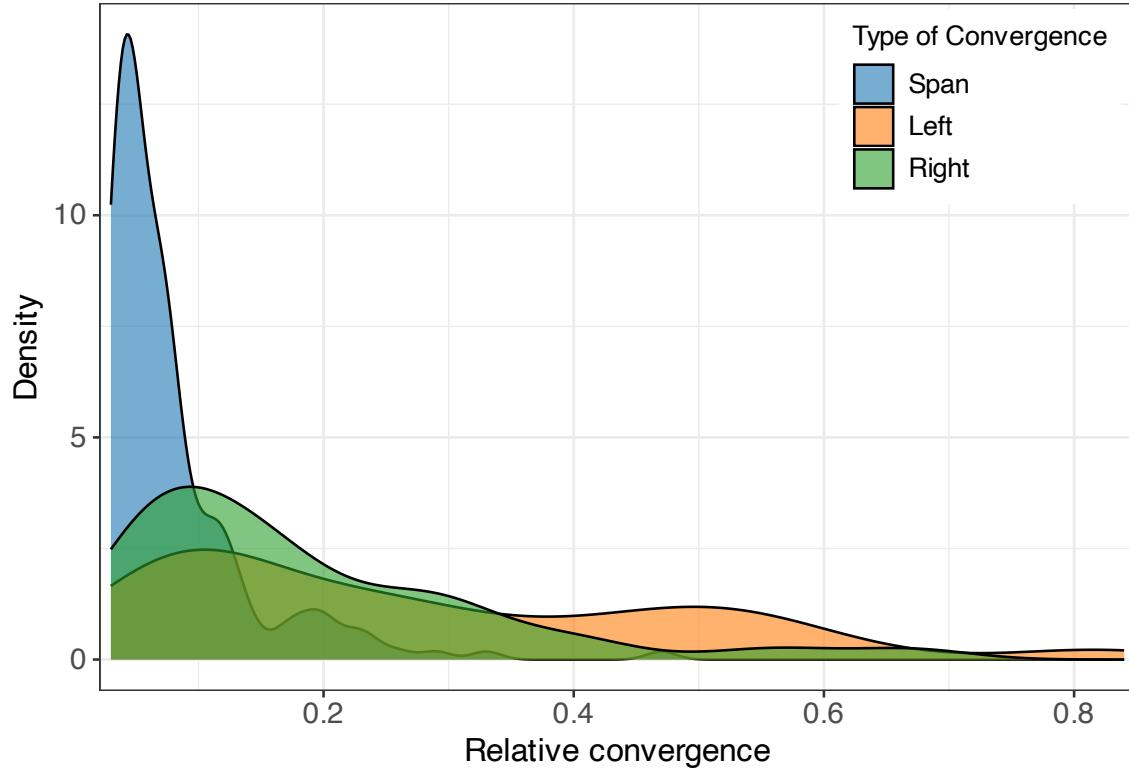
    axis.text = element_text(size = 12),  

    axis.title = element_text(size = 13),  

    strip.text.x = element_text(size = 11)  

)
plot_density_all

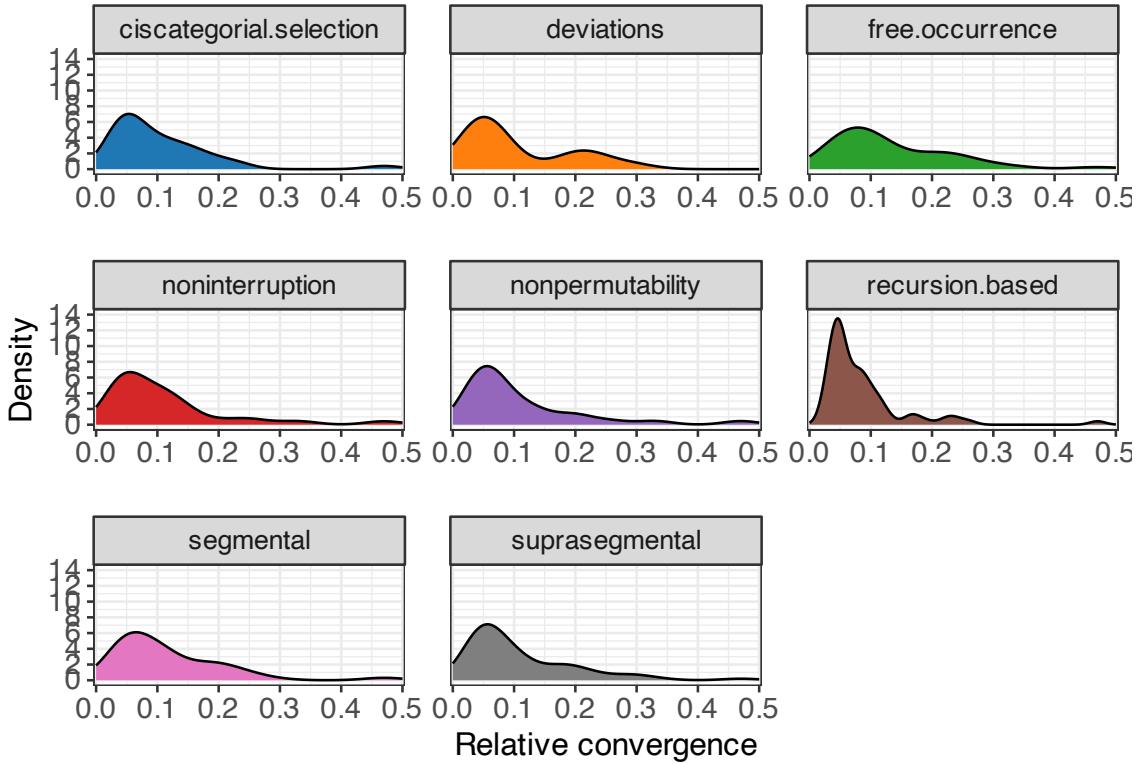
```



Density plots of relative span convergence per abstract type

We do the same, but separating the tests by abstract type. We exclude types with fewer than 5 data points, because these are not informative.

```
# facet density plot of relative convergence per abstract type
plot_dens_conv_atype <- ggplot(aes(x = Relative_Convergence, group = Abstract_Type), data =
= domains_verbal_sub) +
  geom_density(aes(fill = Abstract_Type)) +
  scale_fill_d3(guide = "none") +
  facet_wrap(~Abstract_Type, scales = "free_x") +
  labs(x = "Relative convergence", y = "Density", ) +
  scale_x_continuous(expand = c(0.008, 0), breaks = seq(0, 5, 0.1), limits = c(0, 0.5)) +
  scale_y_continuous(breaks = seq(0, 14, 2), limits = c(0, 14)) +
  theme_bw() +
  theme(
    legend.position = "bottom",
    legend.key.size = unit(1, "lines"),
    legend.text = element_text(size = 11),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 13),
    strip.text.x = element_text(size = 11),
    panel.spacing = unit(1.3, "lines"),
    strip.clip = "off"
  )
plot_dens_conv_atype
```



Density plots of relative span convergence per prosodic word domain

Next, we look at the distributions by type of prosodic word domain. We exclude types with fewer than 5 data points, because these are not informative.

```
# exclude categories with few data points
keep_pw <- domains_verbal_sub %>%
  count(PW_Pattern) %>%
  filter(n > 5) %>%
  droplevels() %>%
  pull(PW_Pattern)

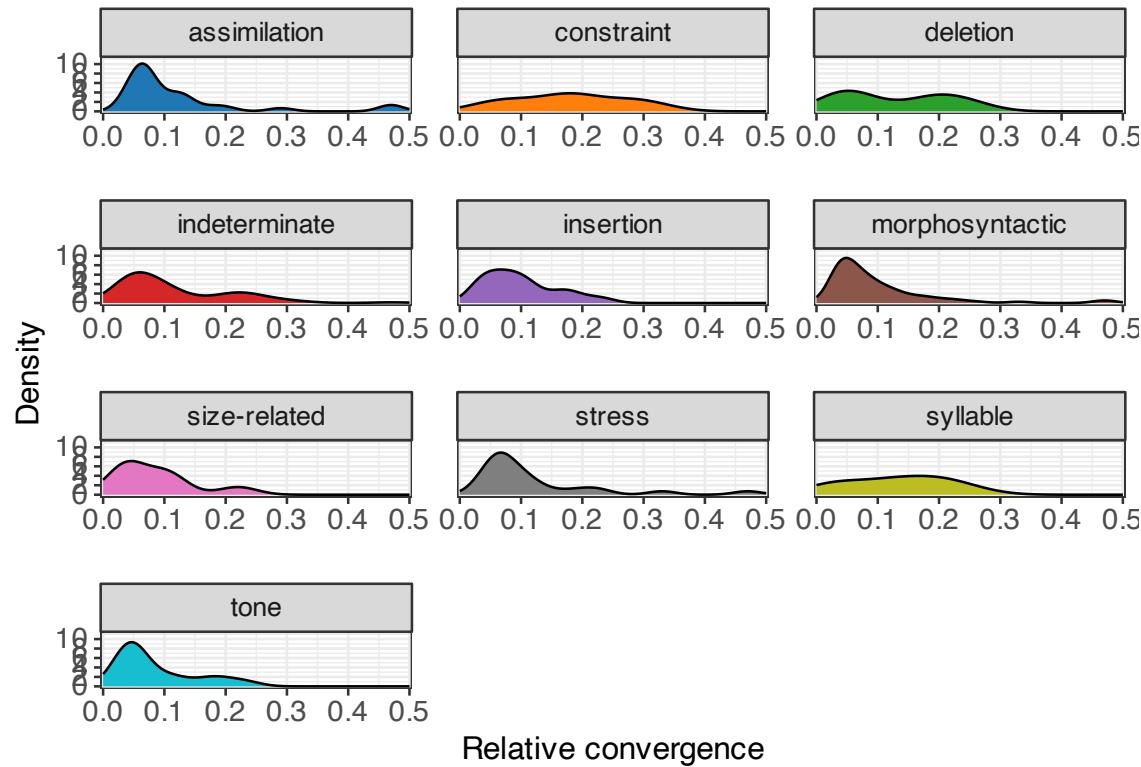
domains_verbal_sub_pw <- domains_verbal_sub %>%
  filter(PW_Pattern %in% keep_pw) %>%
  droplevels()

# make plot
plot_dens_conv_pw <- ggplot(aes(x = Relative_Convergence, group = PW_Pattern), data =
domains_verbal_sub_pw) +
  geom_density(aes(fill = PW_Pattern)) +
  scale_fill_d3(guide = "none") +
  facet_wrap(~PW_Pattern, scales = "free_x", ncol = 3) +
  labs(x = "Relative convergence", y = "Density", ) +
  scale_x_continuous(expand = c(0.008, 0), breaks = seq(0, 5, 0.1), limits = c(0, 0.5)) +
  scale_y_continuous(breaks = seq(0, 11, 2), limits = c(0, 11)) +
  theme_bw() +
  theme(
    legend.position = "bottom",
    legend.key.size = unit(1, "lines"),
```

```

    legend.text = element_text(size = 11),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 13),
    strip.text.x = element_text(size = 11),
    panel.spacing = unit(1.3, "lines"),
    strip.clip = "off"
)
plot_dens_conv_pw

```



Density plots of relative span convergence across cross-language fractures

Lastly, we do the same but look at cross-linguistic fractures with more than 10 tokens each.

```

# take subset with fractures that have more than 10 tokens; exclude NA
keep_clf <- domains_verbal %>%
  filter(!is.na(CrossL_Fracture)) %>%
  count(CrossL_Fracture) %>%
  filter(n > 10) %>%
  droplevels() %>%
  pull(CrossL_Fracture)
domains_verbal_clf <- domains_verbal %>%
  filter(CrossL_Fracture %in% keep_clf) %>%
  droplevels()

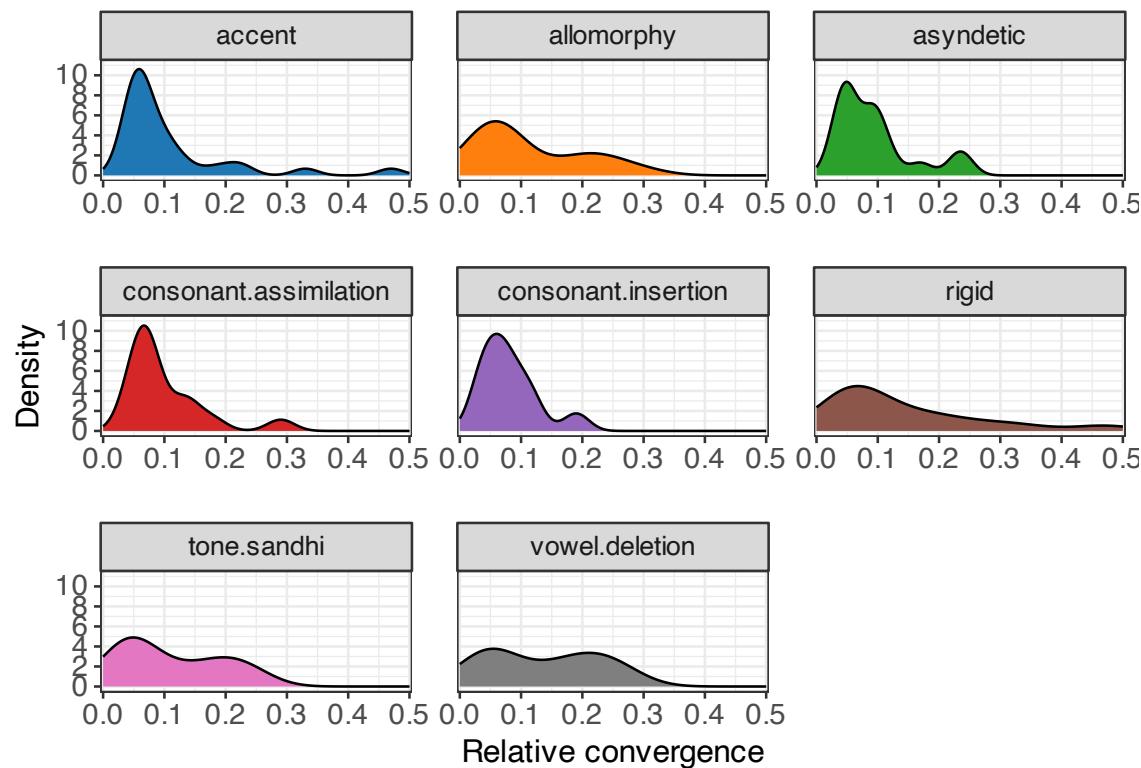
# make plot
plot_dens_conv_clf <- ggplot(aes(x = Relative_Convergence, group = CrossL_Fracture), data =
= domains_verbal_clf) +
  geom_density(aes(fill = CrossL_Fracture)) +
  scale_fill_d3(guide = "none") +

```

```

facet_wrap(~CrossL_Fracture, scales = "free_x", ncol = 3) +
  labs(x = "Relative convergence", y = "Density", ) +
  scale_x_continuous(expand = c(0.008, 0), breaks = seq(0, 5, 0.1), limits = c(0, 0.5)) +
  scale_y_continuous(breaks = seq(0, 11, 2), limits = c(0, 11)) +
  theme_bw() +
  theme(
    legend.position = "bottom",
    legend.key.size = unit(1, "lines"),
    legend.text = element_text(size = 11),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 13),
    strip.text.x = element_text(size = 11),
    panel.spacing = unit(1.3, "lines"),
    strip.clip = "off"
  )
)
plot_dens_conv_clf

```



Random Forest analysis

Section 7

Convergence and span size in phonological domains

To assess the word bisection thesis, i.e. the idea that there are phonological and grammatical words, we have a closer look at convergence and span sizes in phonological vs. morphosyntactic domains per language. We first create the dataset by counting convergences per language with relative span sizes.

```

# count convergences between those domains per language
conv_bisection <- domains %>%

```

```

unite("Spans", Left_Edge:Right_Edge, sep = "-") %>%
group_by(Planar_ID, Domain_Type) %>%
mutate(Total_Tests_D = n()) %>%
group_by(Planar_ID, Domain_Type, Spans) %>%
mutate(DConvergence = n()) %>%
ungroup()

# with indeterminate = morphosyntactic
conv_bisection_lumped <- domains %>%
  unite("Spans", Left_Edge:Right_Edge, sep = "-") %>%
  mutate(Domain_Type_Lumped = if_else(Domain_Type == "indeterminate", "morphosyntactic",
Domain_Type)) %>%
group_by(Planar_ID, Domain_Type) %>%
mutate(Total_Tests_DL = n()) %>%
group_by(Planar_ID, Domain_Type, Spans) %>%
mutate(DLConvergence = n()) %>%
ungroup()

# add back to df for plotting
domains_bisection <- conv_bisection %>%
  left_join(., conv_bisection_lumped) %>%
  mutate(Relative_DConvergence = round(DConvergence / Total_Tests_D, 2),
Relative_DLConvergence = round(DLConvergence / Total_Tests_DL, 2))

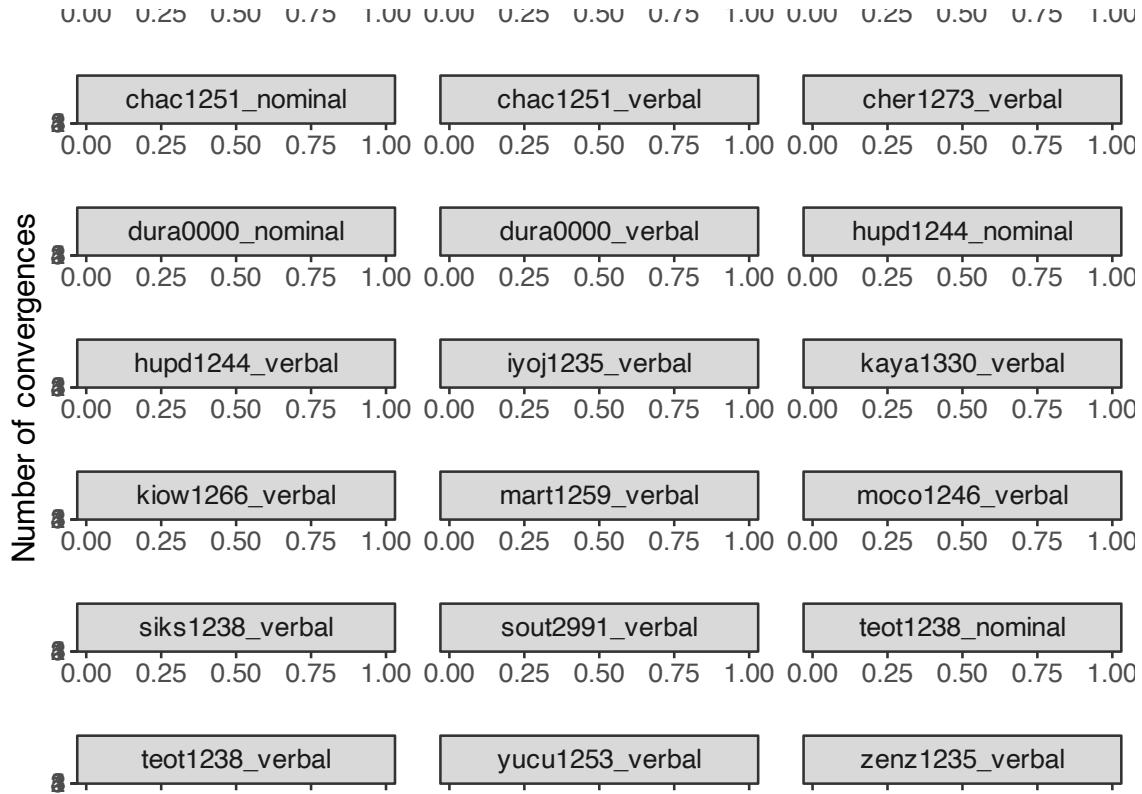
```

We then plot the result for phonological domains.

```

# plot lumped, phonological domains
plot_points_phon <- ggplot(data = filter(domains_bisection, Domain_Type_Lumped ==
"phonological"), aes(x = Relative_Size, y = DLConvergence)) +
  geom_line(lineWidth = 0.5) +
  geom_point(size = 3, color = "darkorange2") +
  facet_wrap(~Planar_ID, scales = "free_x", ncol = 3) +
  labs(x = "Relative span size", y = "Number of convergences") +
  scale_x_continuous(expand = c(0, 0.03), breaks = seq(0, 1, 0.25), limits = c(0, 1)) +
  scale_y_continuous(expand = c(0.1, 0.1)) +
  theme_bw() +
  theme(
    legend.position = "top",
    legend.key.size = unit(1, "lines"),
    legend.text = element_text(size = 11),
    axis.text = element_text(size = 11),
    axis.title = element_text(size = 13),
    strip.text.x = element_text(size = 11),
    panel.spacing = unit(1.3, "lines"),
    strip.clip = "off"
  )
plot_points_phon

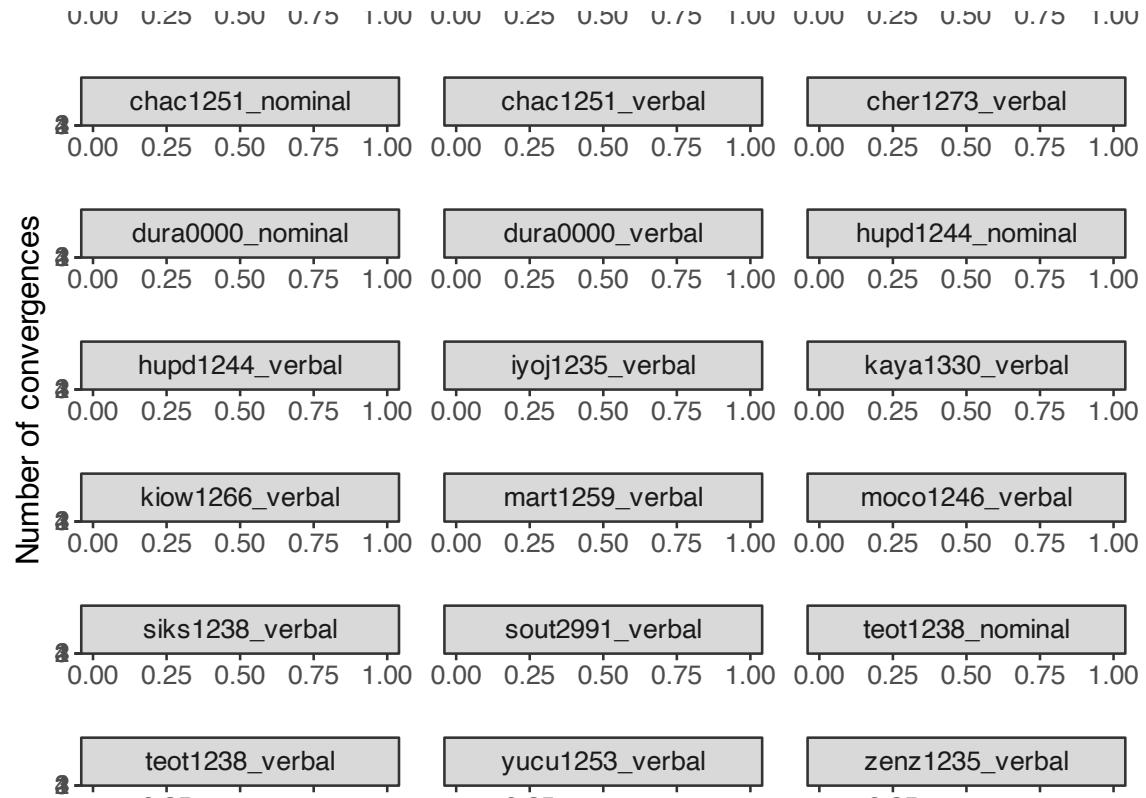
```



Convergence and span size in morphosyntactic domains

And the same for morphosyntactic domains.

```
# plot lumped, morphosyntactic domains
plot_points_ms <- ggplot(data = filter(domains_bisection, Domain_Type_Lumped == "morphosyntactic"), aes(x = Relative_Size, y = DLConvergence)) +
  geom_line(width = 0.5) +
  geom_point(size = 3, color = "dodgerblue3") +
  facet_wrap(~Planar_ID, scales = "free_x", ncol = 3) +
  labs(x = "Relative span size", y = "Number of convergences") +
  scale_x_continuous(expand = c(0, 0.04), breaks = seq(0, 1, 0.25), limits = c(0, 1)) +
  scale_y_continuous(expand = c(0, 0.3)) +
  theme_bw() +
  theme(
    legend.position = "top",
    legend.key.size = unit(1, "lines"),
    legend.text = element_text(size = 11),
    axis.text = element_text(size = 11),
    axis.title = element_text(size = 13),
    strip.text.x = element_text(size = 11),
    panel.spacing = unit(1.3, "lines"),
    strip.clip = "off"
  )
plot_points_ms
```



References

Kahle, David & Wickham, Hadley. 2013. Ggmap: Spatial visualization with ggplot2. *The R Journal*. <https://journal.r-project.org/archive/2013-1/kahle-wickham.pdf> 5(1). 144–161.